Multi-View Action Recognition One Camera At a Time

Scott Spurlock and Richard Souvenir
University of North Carolina at Charlotte
Charlotte, NC 28223 USA
{sspurloc,souvenir}@uncc.edu

Abstract

For human action recognition methods, there is often a trade-off between classification accuracy and computational efficiency. Methods that include 3D information from multiple cameras are often computationally expensive and not suitable for real-time application. 2D, frame-based methods are generally more efficient, but suffer from lower recognition accuracies. In this paper, we present a hybrid keypose-based method that operates in a multi-camera environment, but uses only a single camera at a time. We learn, for each keypose, the relative utility of a particular viewpoint compared with switching to a different available camera in the network for future classification. On a benchmark multi-camera action recognition dataset, our method outperforms approaches that incorporate all available cameras.

1. Introduction

Two significant challenges for human action recognition from a single camera are the unknown relative pose between the camera and person, and the inherent ambiguity of particular actions from certain viewpoints. Distributed camera methods have addressed these challenges by incorporating the information from multiple cameras; however, this data integration introduces new concerns. Compared to single-camera methods, additional cameras in the system, combined with, *e.g.*, 3D models or distributed computing, add to the computational cost. This complexity often improves recognition accuracy, but negatively impacts real-time performance.

In this paper, we present an efficient multi-view action recognition method designed for distributed camera networks. Rather than integrate the information from multiple cameras simultaneously, our method only considers a single camera at each time step, taking advantage of the efficiency of single-camera recognition. The top row of Figure 1 shows a person in a scene captured in a multi-camera network. From the left viewpoint, the pose is ambiguous. However, the right viewpoint represents a discriminative keypose; this image is likely part of a sequence of a person waving. Previous methods [8] take advantage of the observation that certain poses are unique to particular action classes. This work focuses on the observation that, in addition to poses that are discriminative for action classification, certain poses indicate potential future classification ambiguity and can provide information for a *view-shift* to a different camera in the network. In Figure 1, the pose on the left represents such a case.

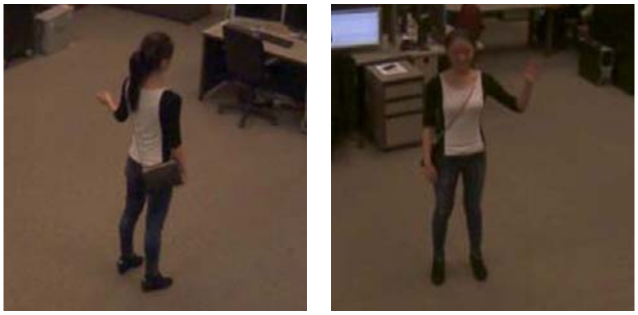

Figure 1. Two frames of the same action captured from multiple cameras. This paper presents an action recognition method that learns if a pose represents a potential future ambiguity (left) and which available viewpoint will facilitate disambiguation (right).

Our method frames view-shift learning as a Markov decision process and employs reinforcement learning to estimate the utility of the available viewpoints in a distributed camera network for human action recognition. Our approach does not require the camera configuration to be the same in training and testing, and, by using a frame-based classification scheme, is applicable to variants of the problem of action recognition, such as early event detection. We compare recognition results of our view-shift method with recent traditional multi-camera approaches and show that our method is comparable to the state of the art on a standard benchmark dataset in terms of recognition accuracy, but with greatly reduced computational effort.

2. Related Work

For multi-view action recognition, many authors (*e.g.*, [14]), distinguish between methods that explicitly build 3D models (*e.g.*, [13, 12]), and methods that may incorporate multiple 2D views, but do not build 3D models. Some of these methods explicitly estimate the viewpoint from which an image was captured (*e.g.*, [10, 6]) and other work follows the approach of designing view-invariant features (*e.g.*, [1, 9]). Our approach is most related to these 2D approaches, but we do not learn to distinguish viewpoints nor rely on view-invariant features. Our method instead predicts if another viewpoint would be advantageous and is agnostic to base feature representation.

In terms of methodology, our approach extracts features frame-wise, rather than across an "action length" video clip. Typically, frame-wise approaches involve quantizing frame-level feature descriptors; a variety of approaches have been proposed. Most similar to our method is the work of Tran and Sorokin [11], who proposed a single-camera classifier using a sequence of quantized keyposes and Naive Bayes classification. Similarly, other methods [16, 3] also weight quantized poses (obtained via supervised or unsupervised methods) by discriminative power and apply an aggregation scheme for action recognition. There are also multi-view extensions that combine camera predictions using voting [7].

The idea of using information from one view to inform view selection can also be found in the area of active vision [5], usually in the context of a mobile agent. For example, in [2], agents perform object recognition using entropy maps, which model the predicted suitability of potential viewpoints to help determine the object. In robotics, this is commonly called the next best view (NBV) problem [4]. The active vision paradigm is not generally applicable to action recognition since the environment or objects of interest are usually static and the time to compute or decide on a view-shift is not a factor, unlike in action recognition.

In Section 3, we describe our classification approach, which uses reinforcement learning to estimate which keyposes indicate a potentially ambiguous classification and how best to view-shift. Our method is applicable to a variety of feature transforms used in action recognition, and in Section 4, we show the high recognition rates of our method on a benchmark dataset.

3. Approach

For training, the input is a set of quantized feature representation (or *keypose*) sequences, $\mathcal{S} = \{S_i\}$ captured from n_C different camera viewpoints. $S_i = \langle s_{i,1}, s_{i,2}, \ldots, s_{i,m} \rangle$ is an m-length sequence of keyposes from the set $X = \{x_j\}$. Each sequence, S_i, is associated with class label, $y_i \in [1, n_L]$, where n_L is the number of class labels. Similar to [11], we compute the posterior probabilities, $P(y \mid S_i)$, using a frame-wise approach with the Naive Bayes and uniform priors assumptions:

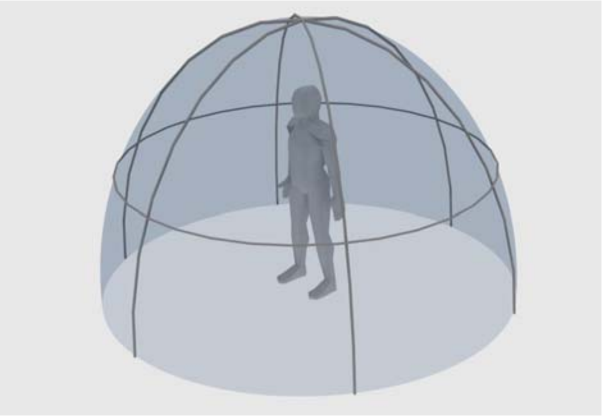

Figure 2. The relative positions of the cameras in a distributed network can be discretized in terms of azimuth and elevation on a half-sphere centered on a target. By considering *relative* view-shifts, our method can learn the utility of view-shifts to viewpoints unseen during training, allowing for flexibility in camera network configuration.

$$P(y \mid S_i) \propto \sum_t \log P(s_{i,t} \mid y). \qquad (1)$$

For single-camera action recognition, the final prediction for sequence, S_i, is the class label, $\hat{y}$, that maximizes the posterior probability:

$$\hat{y} = \underset{y'}{\operatorname{argmax}} \sum_t \log P(s_{i,t} \mid y') \qquad (2)$$

The model for single-camera recognition provides the framework for our multi-view extension with view-shifting. To describe our approach, we first describe our multi-view model and how view-shifts can be learned.

3.1. Multi-View Model

Our method is designed for a network of multiple calibrated cameras. For a given target, the positions of the cameras can be described by a discretized half-sphere centered on the target, indexed by the azimuth and elevation, as depicted in Figure 2. We define a *view-shift* as a change of viewpoint based on a relative offset. The possible view-shifts are determined by the physical location of the cameras relative to the target. To facilitate learning, the half-sphere and, hence, view-shifts, are discretized into a fixed number of azimuth and elevation offsets. Let $\vec{v}$ represent a view-shift in (cyclic) azimuth and elevation. The view-shift, $\vec{v}_0 = \langle 0, 0 \rangle$, represents maintaining the current view.

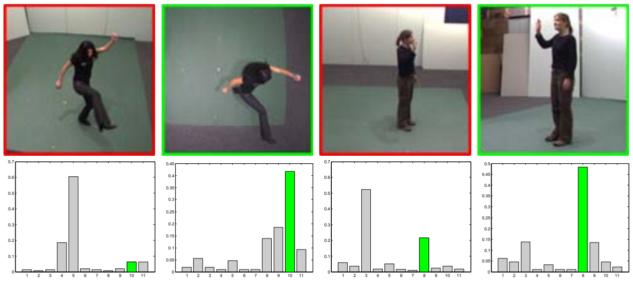

Figure 3. Examples of frames captured from different views and the class probability distributions for each frame with the correct class highlighted in green. In our reinforcement learning method, The frames boxed in green would trigger a positive reward and the red-boxed frames would trigger a negative reward.

3.1.1 Learning View-shifts

Consider the sequence of frames for a particular action. Given an observation (*i.e.*, keypose), at each timestep, the primary view can be shifted to another available view. Our goal is to learn the best view-shift (including maintaining the same view) for each keypose. View-shift learning can be framed as a Markov decision process where the goal is correct action sequence classification. We apply reinforcement learning, specifically Q-learning, to learn the state-action value function, $Q(x, \vec{v})$, which encodes the value of applying view-shift, $\vec{v}$, after observing keypose x. The Q-learning algorithm iteratively approximates $Q(x, \vec{v})$ based on samples taken from training sequences.

$$Q(x,\vec{v}) \leftarrow (1-\eta)Q(x,\vec{v}) + \eta\left[r(x',\vec{v},y^*) + \gamma \max_{\vec{v'}} Q(x',\vec{v'})\right] \tag{3}$$

where η is the learning rate, r is the immediate reward function, x' is the keypose observed after view-shift $\vec{v}$, and y^* is the correct label for training sample. Reward allocation is designed to be proportional to the discriminative power of the next keypose in the sequence with a small penalty for view-shifting. We allocate rewards as follows:

$$r(x,\vec{v},y^*) = \Psi(\vec{v}) + \begin{cases} 1 & \text{if } \operatorname*{argmax}_{y'} P(x \mid y') = y^* \\ -1 & \text{otherwise} \end{cases} \tag{4}$$

where the view-shift penalty, $\Psi(\vec{v}) = -0.1$ if $\vec{v} \neq \langle 0, 0\rangle$. So, we provide a positive reward when the most likely class for the keypose, given by the distribution, $P(x \mid y')$, is the correct class, y^*, and a negative reward otherwise (*i.e.*, the frame is misclassifed). In the next section, we describe the details of the algorithms for training and applying our view-shift recognition method.

3.2. Algorithm

In this section, we describe the design decisions and algorithm details for our method. Our frame-wise approach leverages the discriminative power of individual keyposes, encoded by the independent probability distributions, $P(x \mid y)$, which are computed from the training data using Laplace smoothing:

$$P(x \mid y) = \frac{n_K + 1}{n_A + |X|} \tag{5}$$

where n_K is the number of instances in the training data where keypose x occurs in sequences with label y, n_A is the total number of training frames with class label y, and $|X|$ is the number of distinct keyposes. Figure 3 shows examples of two frames captured from different views and the class probability distributions, $P(x \mid y)$, for each frame with the correct class highlighted in green. Based on the scoring scheme, in both cases, the pose on the right (boxed in green) represents a positive reward and that on the left (boxed in red) represents a negative reward. The view-shift *training* algorithm is summarized below:

1. For each training frame, extract and quantize features for keypose representation.
2. For each keypose, calculate frame conditional probabilities (Equation 5).
3. Initialize $Q(x, \vec{v})$ for all keyposes and view-shifts.
4. Repeat until convergence
   (a) Randomly select a training action sequence S_i (with associated view, C_i).
   (b) For each keypose, $x = s_{i,t}$, in the sequence
      i. For each available view-shift, compute reward (Equation 4).
      ii. Update Q using Equation 3.

The result of this training process is the state-action value function, $Q(x, \vec{v})$, which represents the value of each view-shift on the half-sphere for each keypose in the dictionary. A test sequence consists of single-camera input with potential alternative viewpoints for view-shifts. We modify the single-camera Naive Bayes classifier for view-shifting *recognition* as follows:

1. Repeat until classification criterion is met
2. Extract and quantize features for keypose representation.
3. Update classification posteriors (Equation 2).
4. Look up the values, $Q(x, \vec{v'})$, for each available view-shift, $\vec{v'}$.
5. Switch to the camera corresponding to the maximum-value view-shift.

For real-time action recognition, there are many possibilities for reporting a prediction, including time- or confidence-based classification. In the next section, we show results for a variety of classification criteria.

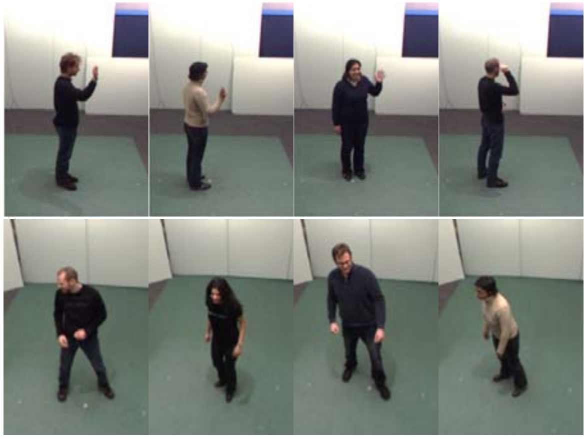

Figure 4. Each row shows frames co-clustered after unsupervised dictionary learning. (Top) The cluster is mostly homogeneous; most of the frames are examples of waving. (Bottom) The cluster is heterogeneous. From left to right, the actions are pick-up, kick, punch, and check watch.

4. Results

We evaluate our approach using the INRIA Xmas Motion Acquisition Sequences (IXMAS) dataset [13], a standard benchmark for multi-view action recognition, which contains multiple actors performing various actions, captured by five synchronized, calibrated cameras. The position and orientation (relative to the cameras) of the IXMAS actors are not prescribed, so the relative pose of an actor is not a function of which camera is recording the action sequence. All of the methods described in this section were implemented in Matlab on a standard PC. Our approach can be applied to any frame-based feature descriptor; our experiments use the Motion Context [11] descriptor, which represents the distribution of occupancy and $x-$ and $y-$ components of optic flow in a bounding box surrounding the object of interest combined with a low-dimensional projection of the feature vectors for neighboring frames.

Keypose Learning To build the dictionary of keyposes, the feature descriptors for every fourth frame in the training data were clustered via the k-means algorithm. For these experiments, k was set to 150 clusters per class. We initialized k-means 5 times and selected the cluster assignment with minimum energy, as measured by average intra-class similarity. Figure 4 shows example frames from two clusters. The first row depicts a cluster that is mainly homogeneous; most of the frames are examples of waving. The second row shows the more common case of a pose shared among multiple classes.

View-shift Learning IXMAS consists of five cameras. Four of the cameras are at similar elevations, and one is nearly overhead. For this configuration, we discretize the view-sphere into two bins for elevation and 10 for azimuth, as shown in Figure 5. This results in 30 possible viewshifts: 10 for clockwise shifts in azimuth, and 3 for elevation (+1, 0, -1). For the experiments described in the next section, the number of keyposes (*i.e.*, dictionary size) is ~1500. Therefore, the state-action value table has ~50k entries. For Q-learning, the free parameters were determined empirically. The discount factor, γ, is 0.5, and the learning rate, η, is initially 1.0 with a decay rate of .997. In addition to the reward allocation scheme represented in Equation 4, we evaluated other schemes, including rewards proportional to the keypose classification margin or based on logistic regression; in general, the selected scheme performed at least as well as these alternatives. Training terminates when 10 successive entries are updated by less than 10^{-5}. Training for a typical experiment using IXMAS data with the motion context feature requires ~5000 iterations to converge, which takes ~30 seconds on a standard PC. Figure 6 shows an example of a view-shift learned by this method. For this sequence, the pose represented by the third frame (outlined in green) induces a view-shift to a camera with views more discriminative for the anticipated ambiguity.

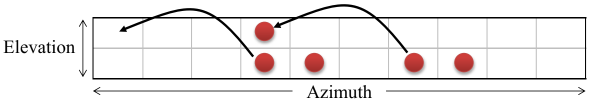

Figure 5. For the 5-camera IXMAS dataset, this diagram shows the locations of each the camera (red circles) when the view half-sphere is discretized into 10 azimuth and 2 elevation bins. The arrows represent a viewshift; both arrows indicate the same view-shift: $\langle +3, -1 \rangle$. Relative view-shifts allow our method to learn the utility of view-shifts to viewpoints unavailable during training.

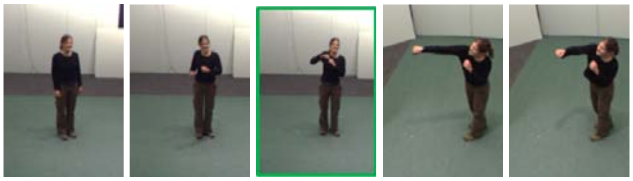

Figure 6. In this punching sequence, the pose in the third frame, outlined in green, is ambiguous and represents potential future confusion from this viewpoint. The ambiguous pose triggers a learned view-shift. The final two frames, viewed from the shifted camera, are highly discriminative for this action.

4.1. Experiments

We apply our method to a few variants of the action recognition problem and demonstrate the robustness of our method to different camera configurations in training compared to the testing environment. We refer to our view-shift action recognition method as `vs` and compare to alternative frame-based multi-view classification schemes that use the

Method	type	Accuracy
`vs`	single	**94.24%**
`vote`	multi	93.33%
`sc`	single	86.06%
`wvote`	multi	83.33%
Liu et al. [7]	multi	93.7%
Wu et al. [15]	multi	88.2%
Zhu et al. [17]	multi	88.0%

Table 1. Multi-view classification rates on the IXMAS data set. The method type refers to the number of simultaneous views. (Top) Rates for our approach, `vs`, and variants. (Bottom) Representative 2D multi-view recognition rates reported in the literature.

same features and aggregation methods:

- The single-camera (`sc`) method is our implementation of an algorithm described in [11], which uses Naive Bayes classification on a sequence of frames without view-shifting.
- The multi-camera voting (`vote`) method applies a common multi-view aggregation technique where the majority decision serves as the final classification.
- The weighted multi-camera voting (`wvote`) extends (`vote`) by weighting each vote by the normalized posteriors returned by the classifier.

Action Recognition The first experiment is the most common action recognition task: classification on video sequences of prescribed length. We followed the experimental protocol most commonly found in the literature for this dataset (*e.g.*, [13, 17, 7]), which uses 10 actors, each performing one of 11 actions three times and is evaluated with a leave-one-actor-out cross validation strategy. Table 1 shows the overall accuracy of each of the methods and also a comparison to other recent 2D multi-view recognition methods on this dataset. Our view-shift method, (`vs`), achieves an accuracy of 94.24%, which, to the best of our knowledge, is the highest accuracy achieved on this dataset among 2D multi-view methods. Compared to `vote` and the top-performing methods from the literature ([7, 17]), our approach not only achieves higher accuracy, but is more efficient (22 fps compared to 4.5 fps) since, per target, processing occurs on only a single view per time step. The `wvote` method performed worse than the single-camera method, possibly due to the unreliability of Naive Bayes posterior probabilities as estimators.

Figure 7 shows the confusion matrix for our method, `vs`, for this classification experiment on the IXMAS data set. Each row represents the actual class and each column represents the predicted class. For many actions (*e.g.*, sit, stand, walk, kick, pickup), accuracy is 100%. The most challenging case involves confusion between waving and scratching head. This is reasonable as the base features are mainly silhouette-based and these actions include self-occlusion from most views.

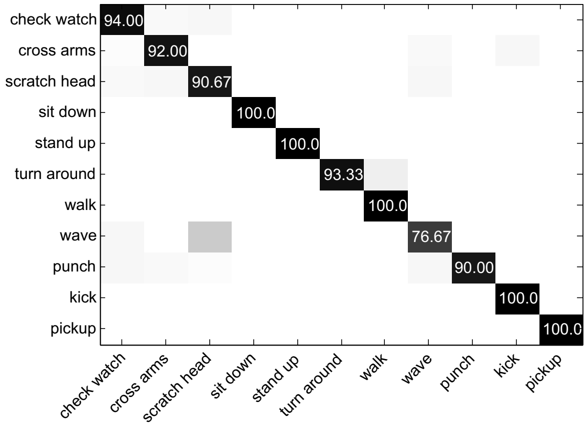

Figure 7. Confusion matrix for the proposed view-shift method on the IXMAS dataset.

Early Recognition In this variant of action recognition, the task is to classify the action prior to the end of the sequence. Previous work (*e.g.*, [8]) has included the observation that many action sequences can be identified with snippets of only a few frames. Our view-shift method is frame-based and capable of early classification without modification. For this experiment, we classified only the first portion of an action sequence for various lengths from 10% to 100%. Figure 8 shows the results for our method and the representative single- and multi-view methods. As expected, the classification accuracy of all methods increase as a greater portion of each sequence is observed. As with the experiment with the full sequences, our method achieves comparable performance to the method that incorporates all views simultaneously. For sequences comprising just the first 40% of the available frames, the view-shift method accuracy is 85%, which is competitive with recent methods that observe the entire sequence and utilize all the cameras simultaneously (as reported in Table 1).

Camera Network Configuration The previous experiments operate under the assumption that the configurations of the training and testing environments are the same. In a real-world setting, this may not be the case. Our method learns relative view-shifts, rather than a model specific to a particular camera network, and can be applied to camera configurations not used in training. We perform experiments using various combinations of the cameras in the IXMAS dataset for training and testing. Table 2 shows results for an experiment where four of five cameras were used in training, and all five cameras were used for testing. For single-camera recognition (`sc`), accuracy drops by an average of 12% when recognizing sequences that include a

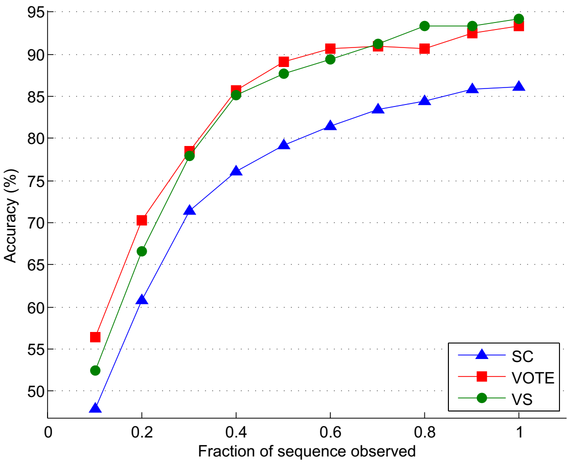

Figure 8. Accuracy of action recognition as a function of the observed fraction of each sequence. As the fraction increases, the accuracy of all methods improves.

Training Views	sc	vote	vs
2 3 4 5	77.70%	91.52%	92.00%
1 3 4 5	74.79%	91.82%	92.12%
1 2 4 5	76.06%	90.61%	91.70%
1 2 3 5	75.52%	92.12%	91.94%
1 2 3 4	74.36%	90.61%	91.09%
Average	75.69%	91.34%	91.77%

Table 2. Classification accuracy on the IXMAS dataset with various camera combinations. The left column shows which cameras were used for training.

camera unseen during training. By contrast, with the view-shift method, vs, recognition accuracy does not significantly drop when the test environment contains a viewpoint unseen in training. As before, our method achieves similar performance to the more computationally expensive approach, vote, which uses multiple views simultaneously. Additionally, we perform an experiment using fewer cameras in testing compared to training. We train the method using all five cameras, and perform recognition using a subset of these views. Averaging the results over all the permutations of cameras, our method, vs, achieves 91.93%, 90.66%, 90.33% for four, three, and two cameras, respectively. As the number of cameras in the test environment decreases, the accuracy of our method decreases only slightly. While these experiments do not replicate disparate camera networks, they provide evidence that relative view-shifts are not tied to specific camera configurations.

5. Conclusions and Future Work

In this paper, we presented a new approach to multi-camera action recognition that makes use of only a single active camera at a time, resulting in computational efficiency, while achieving results equal to or slightly better than methods that incorporate multiple views simultaneously. In the future, we plan to deploy the method on a distributed camera network in a real-world setting for real-time action recognition. We will evaluate feature representations best-suited to real-time performance and consider reward allocation schemes in our learning apporach that more closely correspond to the (potentially variable) real-world costs of dynamic camera selection.

References

[1] S. Ali and M. Shah. Human action recognition in videos using kinematic features and multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(2):288–303, 2010.

[2] T. Arbel and F. Ferrie. Viewpoint selection by navigation through entropy maps. In *Proc. Intl Conf. on Computer Vision*, volume 1, pages 248–254, 1999.

[3] S. Cheema, A. Eweiwi, C. Thurau, and C. Bauckhage. Action recognition by learning discriminative key poses. In *IEEE Intl Conf. on Computer Vision Workshops*, pages 1302–1309, 2011.

[4] E. Dunn and J. Frahm. Next best view planning for active model improvement. In *Proc. of the British Machine Vision Conf.*, 2009.

[5] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Active recognition through next view planning: a survey. *Pattern Recognition*, 37(3):429–446, 2004.

[6] A. Farhadi, M. Tabrizi, I. Endres, and D. Forsyth. A latent model of discriminative aspect. In *IEEE Intl Conf. on Computer Vision*, pages 948–955, 2009.

[7] L. Liu, L. Shao, and P. Rockett. Boosted key-frame selection and correlated pyramidal motion-feature representation for human action recognition. *Pattern Recognition*, 46(7):1810 – 1818, 2013.

[8] K. Schindler and L. Van Gool. Action snippets: How many frames does human action recognition require? In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[9] A. Sharma, A. Kumar, H. Daume, and D. Jacobs. Generalized multiview analysis: A discriminative latent space. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2160–2167, 2012.

[10] R. Souvenir and J. Babbs. Learning the viewpoint manifold for action recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–7, 2008.

[11] D. Tran and A. Sorokin. Human activity recognition with metric learning. In *European Conf. on Computer Vision*, pages 548–561, 2008.

[12] P. Turaga, A. Veeraraghavan, and R. Chellappa. Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[13] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104(2):249–257, 2006.

[14] D. Weinland, R. Ronfard, and E. Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, 115(2):224–241, 2011.

[15] X. Wu, D. Xu, L. Duan, and J. Luo. Action recognition using context and appearance distribution features. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 489–496, 2011.

[16] Z. Zhao and A. M. Elgammal. Information theoretic key frame selection for action recognition. In *British Machine Vision Conf.*, pages 1–10, 2008.

[17] F. Zhu, L. Shao, and M. Lin. Multi-view action recognition using local similarity random forests and sensor fusion. *Pattern Recognition Letters*, 2012.